CAAM 519: Computational Science I
Fall 2015 Syllabus

Matthew G. Knepley
3021 Duncan Hall knepley@rice.edu

This course concerns scientific computation in a broad sense. We will use numerical libraries written in higher level languages, such as C, Fortran, and Python, to solve computational problems. Emphasis will be placed on computational decision making. For example, how should I choose which nonlinear solver to use? How should I evaluate its suitability once I have tried it? We will also cover basic techniques of algorithm design and implementation, project planning, source management, configuration and build tools, documentation, program construction, i/o, and visualization.

If you go on to career in scientific computing, you will very often be confronted with models, discretizations, numerical methods, and solution algorithms unfamiliar to you. Some you will research and understand completely, and some you will understand at a merely mechanical level. This course mirrors that experience. When you first did arthmetic, were you aware you were doing it in an algebraically closed field or why?

1 Course Information

Instructor	Matthew G. Knepley
Class times	11am to 11:50am on Monday, Wednesday, & Friday
Location	Duncan Hall 1042

A course overview and grading policy are available in accordance with Rice academic policy.

2 Required and Recommended Reading

Class notes have been prepared for each lecture. Students should read the notes prior to attending class, but the lecture may deviate from the notes somewhat. The notes will be continually updated as the class progresses, so students should download them again at the beginning of each week. The lectures for each week of class are contained in a chapter. The weekly lectures will roughly follow the chapter sections.

The following books are foundational for the software construction and documentation will be attempt in this class. However we will not follow them in any way, and they are intended for reference and outside reading.

- The C Programming Language (2nd Edition) by Brian W. Kernighan and Dennis M. Ritchie (Prentice Hall) ISBN-10: 0131103628.
- The PETSc Manual, S. Balay, S. Abhyankar, M. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. Gropp, D. Kaushik, M. Knepley, L. Curfman McInnes, K. Rupp, B. Smith, and H. Zhang, ANL-95/11 Rev 3.5, 2015.
- LaTex Companion (Tools and Techniques for Computer Type Setting), 2nd ed., by Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, and Chris Rowley (Addison Wesley Professional) ISBN-10: 0201362996.

3 Schedule

3.1 Week 1: Programming Basics

3.1.1 Lecture 1: Scientific Computing

The paper is the perfect vehicle for mathematical communication, but not for communication of results in scientific computing. In this arena, it is the numerical library.

Online resources

- The Case Against Credentialism by James Fallows.

3.1.2 Lecture 2: Git

We discuss distributed version control systems, and in particular Git.

Online resources

- Really fantastic graphical, interactive Git Branching Tutorial
- Online Git Documentation
- 10 Years of Git

3.1.3 Lecture 3: C, Make, Python, Valgrind

We discuss configure, compiling and linking C code, makefiles, Python, and using Valgrind for debugging.

Online resources

- C Information
  - GNU C Programming Tutorial, which is very extensive

  - Learn C The Hard Way, which includes linking
  - Programming Question Site
  - Wiki entry on Iso C
  - The C standard library
  - C Frequently Asked Questions: useful to look up particular questions
  - The C book
  - C Programming Tutorial
  - How C programming works (from How stuff works)
  - A short essay about the origins of the C programming language and why it became so popular so quickly
  - The printf command explained concisely and including all bits and pieces
- Ccache, a fast C/C++ compiler cache
- GNU Make Manual
- Valgrind, a memory debugger and much more
- Main Python Site

3.2 Week 2: Finding and Relating Information

3.2.1 Lecture 4: Self-Teaching

The class learns to ask all questions online immediately. Google Omnia Responsit.

Online resources

- Search
  - The 800lb Gorilla, where you can search for error messages. It also effectively indexes mailing lists and question boards.
  - For all papers and other publications, use Google Scholar since it also gives BibTeX entries.
  - Wikipedia is far and away the best mathematics resource on the web
  - Online Encyclopedia of Integer Sequences by Ian Sloane
  - Digital Library of Mathematical Functions by NIST
  - The Mathematics Genealogy Project is the best place to trace academic lineages
- Stack Exchanges (best place for questions and collaborative discussion)
  - Scientific Computing
  - Programming
  - Mathematics
  - TeX

3.2.2 Lecture 5: TeX and LaTeX

We introduce LaTeX and its use in this course.

Online resources

- Installing LaTeX anywhere
- TeX Question Site

3.2.3 Lecture 6: Cancelled

I will be out of town at a conference.

3.3 Week 3: PETSc Introduction

3.3.1 Lecture 7: Numerical Linear Algebra

We become acquainted with PETSc's Vec and Mat.

3.3.2 Lecture 8: Debugging and Performance Monitoring

We learn how to use the debugger properly, and how to interpret a simple log summary.

3.3.3 Lecture 9: PETSc Design

Why don't we use C++? Why don't we use Python? Why don't we use Julia? Why don't we use templates? What would we like to do?

Online resources

- Programming Languages for Scientific Computing

3.4 Week 4: Parallelism

3.4.1 Lecture 10: Basics of MPI

Basics of MPI, and design of layering (Communicators and attributes), running in parallel

Online Resources

- MPI Main Site and method documentation
- MPI Book
- Rice Research Computing and cluster documentation
- Jeff Hammond's talk on MPI+MPI

3.4.2 Lecture 11: Scaling

Definitions of strong and weak scaling. Design of tests. Discussion of what they measure.

3.4.3 Lecture 12: Heterogeneous Parallelism

Arguments for MPI+MPI, and parts of MPI-3 students should learn about.

3.5 Week 5: Data Layout and Discretization I

3.5.1 Lecture 13: Sparse Data

We look at data structures for sparse data, and especially for sparse matrices. We examine the COO, AIJ (looks like run length encoding for 0), and ELL.

Online Resources

- Iterative Methods for Sparse Linear Systems by Yousef Saad

3.5.2 Lecture 14: Generic Data Layout

We look at how PetscSection encodes a data layout, and give many examples.

3.5.3 Lecture 15: What We Do Not Understand

We take a breather to ask questions and go over some of the material already covered.

3.6 Week 6: Simple Finite Differences

3.6.1 Lecture 16: Structured Meshes

Explain DMDA

3.6.2 Lecture 17: Residual Evaluation

Explain ex5 and ex19

3.6.3 Lecture 18: Checking the Solution

Proto-MMS, looking at pointwise convergence

3.7 Week 7: Data Layout and Discretization II

3.7.1 Lecture 19: Unstructured Meshes

We discuss the basics of the Plex interface for unstructure meshes.

Online Resources

- Mesh Algorithms for PDE with Sieve I: Mesh Distribution, Matthew G. Knepley and Dmitry A. Karpeev.

3.7.2 Lecture 20: Unstructured Data Layout

We learn about data layout for mesh data, geometry (several kinds, like Bezier), solution data, adjacency data, etc.

3.7.3 Lecture 21: Parallel Operations

We discuss composable methods of parallel data distribution and redistribution.
Online Resources

- Unstructured Overlapping Mesh Distribution in Parallel, Matthew G. Knepley, Michael Lange, and Gerard J. Gorman

3.8 Week 8: Simple Finite Elements

3.8.1 Lecture 22: Residual Evaluation

Basic steps for finite element residual computation

3.8.2 Lecture 23: Method of Manufactured Solutions (MMS)

Precise steps for MMS. Discussion of fine points. Possible incorporation of Sympy.

3.8.3 Lecture 24: When FEM goes Wrong

Look closely at May review. L_2 vs. l_2 comparisons. The consequences of unstable element pairs, even if we do not understand stability.

3.9 Week 9: Simple Performance Models

3.9.1 Lecture 25: Basic Performance Modeling

We learn why performance modeling is necessary.
Online Resources

- The STREAM benchmark from John McCalpin
- PETSc FAQ on Performance

3.9.2 Lecture 26: The Roofline Model and Gaming the System

We learn about the popular Roofline model.
Online Resources

- Roofline model and talk

3.9.3 Lecture 27: Sparse Matrix-Vector Product (SpMV)

We understand how well SpMV can perform in theory.
Online Resources

- Towards Realistic Performance Bounds for Implicit CFD Codes, by William D. Gropp, Dinesh K. Kaushik, David E. Keyes, and Barry F. Smith

3.10 Week 10: PETSc Linear Equation Solvers

We learn how to solve many equations at once, even if they are only linear.

3.10.1 Lecture 28: Krylov Methods

We explore Krylov solvers computationally. We discuss the main differences without proof. We look at the Nachitgal paper showing computationally that it is hard to pick a solver. We look at the Greenbaum paper showing that it is very hard to predict the behavior of GMRES.

Discussion: List the reasons that solving Navier-Stokes with a Krylov method is a bad idea.

- It is not fundamentally scalable in problem size
- Segregated sovlers from the 60s perform BETTER!
- We have optimal solvers (block solvers, MG with special smoothers)

Online Resources

- Iterative methods for sparse linear systems by Yousef Saad
- An Introduction to the Conjugate Gradient Method Without the Agonizing Pain by Jon Shewchuk

3.10.2 Lecture 29: Direct Solvers

We look at LU, Cholesky

3.10.3 Lecture 30: Multigrid

Basic computational steps in MG, and hand-waving explanation for its effectiveness. Distinction between classical MG and FAS. Simple proof using Taylor series from Barry.

3.11 Week 11: PETSc Nonlinear Equation Solvers

3.11.1 Lecture 31: Single-step Nonlinear Solvers

We learn how to solve systems of nonlinear equations. Nonlinear Richarson, Newton, Picard, Nonlinear Conjugate Gradients, etc.

3.11.2 Lecture 32: Multistep Nonlinear Solvers

Nonlinear GMRES, Anderson Acceleration, Multipoint methods from Traub, etc.

3.11.3 Lecture 33: Nonlinear Preconditioning

Composition of nonlinear solvers is defined, and we explore its use in model problems from the paper.

Online Resources

- Iterative Methods for Nonlinear Systems, Werner Rheinboldt
- Convergence Analysis for Anderson Acceleration, A. Toth and C. T. Kelley
- Composing Scalable Nonlinear Algebraic Solvers, Peter R. Brune, Matthew G. Knepley, Barry F. Smith and Xuemin Tu

3.12 Week 12: Performance Models of Real Applications

3.12.1 Lecture 34: What is different with applications?

3.12.2 Lecture 35: PyLith

Online Resources

- A Domain Decomposition Approach to Implementing Fault Slip in Finite-Element Models of Quasi-static and Dynamic Crustal Deformation, Brad T. Aagaard, Matthew G. Knepley, and Charles A. Williams

3.12.3 Lecture 36: pTatin3d

Online Resources

- pTatin3D: High-Performance Methods for Long-Term Lithospheric Dynamics, Dave A. May, Jed Brown, and Laetitia Le Pourhiet

3.13 Week 13: Interface Design

3.13.1 Lecture 37: Block Preconditioners

We show the simplicity and power of the PCFIELDSPLIT interface, arising from composability.

3.13.2 Lecture 38: Multigrid Preconditioners

We show the simplicity and power of the PCMG interface, arising from composability.

3.13.3 Lecture 39: Composability

We show examples of robust solvers constructed by composing existing strategies.

3.14 Week 14: FEniCS Challenge

3.14.1 Lecture 40: Problem Definition

We define the FEniCS Challenge problem and discuss its relevance to the wider scientific computing community.

3.14.2 Lecture 41: Performance Estimates

We make simple estimates of performance for competing strategies, and get some initial data.

3.14.3 Lecture 42: Final Assessment

We collect all results and determine a winner.

3.15 Week 14: Guest Week (Alternate)

3.15.1 Lecture 40: Tobin Issac

Non-Conforming Meshes

3.15.2 Lecture 41: Jed Brown

Performance Flexibility?

3.15.3 Lecture 42: Anders Logg

Cut Mesh Methods